International Journal on Web Service Computing (IJWSC), Vol.1, No.2, December 2010

BUSINESS RULE MANAGEMENT FRAMEWORK FOR ENTERPRISE WEB SERVICES

Thirumaran. M[1] and Ilavarasan. E[2] and Thanigaivel. K[3] and Abarna. S[4]

[1]Department of Computer science and Engineering, Pondicherry Engg College, India.
thirumaran@pec.edu

[2]Department of Computer science and Engineering, Pondicherry Engg College, India.
eilavarasan@pec.edu

[3]Department of Computer science and Engineering, Pondicherry Engg College, India.
thanigaivel20@pec.edu

[4]Department of Computer science and Engineering, Pondicherry Engg College, India.
abarna.pec@gmail.com

1. ABSTRACT

Making a business rule extraction more dynamic is an open issue, and we think it is feasible if we decompose the business process structure in a set of rules, each of them representing a transition of the business process. As a consequence the business process engine can be realized by reusing and integrating an existing Rule Engine. We are proposing a way for extracting the business rules and then to modify it at the runtime. Business rules specifies the constraints that affect the behaviors and also specifies the derivation of conditions that affect the execution flow. The rules can be extracted from use cases, specifications or system code. But since not many enterprises capture their business rules in a structured, explicit form like documents or implicit software codes, they need to be identified first, before being captured and managed. These rules change more often than the processes themselves, but changing and managing business rules is a complex task beyond the abilities of most business analysts. The capturing process focuses on the identification of the potential business rules sources. As business logic requirements change, business analysts can update the business logic without enlisting the aid of the IT staff. The new logic is immediately available to all client applications. In current trend the rules are modified or changed in the static time phase. But this paper provides to change the rules at the run time. Here the rules are extracted from the services and can be a changed dynamically. The existing rules are modified and attached to source code without hindering service to the end user which can be achieved with source control systems. When the rules are revised, it provides a path in budding new business logic. This new business logic can be adopted for the efficient software development.

2. KEYWORDS

Dynamic Rule Execution, Business process, Business rules, Business Rules Approaches.

3. INTRODUCTION

A Business Rule Management System is a software system used to define, deploy, execute, monitor and maintain the variety and complexity of decision logic that is used by operational systems within an organization or enterprise. This logic, also referred to as business rules, includes policies, requirements, and conditional statements that are used to determine the tactical actions that take place in applications and systems. A business rule management system

(BRMS) enables organizational policies – and the operational decisions associated with those policies – to be defined, deployed, monitored and maintained separately from core application code. By externalizing business rules and providing tools to manage them, a BRMS allows business experts to define and maintain the decisions that guide systems behavior, reducing the amount of time and effort required to update production systems, and increasing the organization's ability to respond to changes in the business environment. BRMS solutions automate policies in custom and composite business applications. They lower application maintenance costs, facilitate more accurate and consistent business policy implementation across applications, and improve collaboration between your business and IT departments.

A business process defines the context and the logical relationships between activities, and also specifies both the order of invocations (control flow) and the rules for data transfer (data flow). Such activities are configured in order to produce a specific output and associated with specific objectives. A business rule is a statement that defines or constrains some aspect of the business. Business rules are usually expressed either as constraints or in the form "if condition then action". They provide the means to express, manage and update different kind of components contained into the whole business domain. In this way, such rules have to be expressed in terms of the defined activities and further integrated with the process specification. The definition of a process and business rules requires often specific skills, usually related to people with a high level of expertise in design and programming. Another common problem is the lack of flexibility related to: business rules derivation based on dynamic changes of the business context. On the other hand, we have seen that many approaches are not completely dynamic and effective when we need to automatically modify the business process instance, by adding or deleting an activity, according to changes of business process context. To address these issues, we propose an approach focused on providing a more dynamic and flexible adaptation of the business rules. The main purpose of our approach is to increase the adaptability of the system, by identify and change or alter the rules in the run time of any application. This can be implemented with the help of JESS language. The JESS language supports dynamic changes that can be made in any real time application.

4. LITERATURE SURVEY

Business Process Management (BPM) is an established discipline for building, maintaining, and evolving large enterprise systems on the basis of business process models. A business process model is a flow-oriented representation of a set of work practices aimed at achieving a goal, such as processing a customer request or complaint, satisfying a regulatory requirement, etc. The Business Process Modeling Notation (BPMN) is gaining adoption as a standard notation for capturing business processes. The main purpose of business process models generally, and BPMN models in particular, is to facilitate communication between domain analysts and to support decision-making based on techniques such as cost analysis, scenario analysis, and simulation Chun Ouyang [1] proposed an acyclic BPMN model, or an acyclic fragment of a BPMN model, falls under the class if it satisfies a number of semantic conditions such as absence of deadlock. He applied Petri net analysis techniques to statically check these semantic conditions on the source BPMN model. According to Michael zur Muehlen [2], Business Processes are sets of activities that create value for a customer. While research in Business Process Management initially focused on the documentation and organizational governance of processes, organizations are increasingly automating processes using workflow systems, and are building elaborate management systems around their processes. Such management infrastructures integrate modeling, automation, and business intelligence applications. The inclusion of compliance management activities is a logical next

step in governing the business process life cycle. Josef Schiefer [3] says that Business Process Management (BPM) systems are software solutions that support the management of the life cycle of a business process. For the execution of business processes, many organizations are increasingly using process engines supporting standard-based process models (such as WSBPEL) to improve the efficiency of their processes and keep the testing independent from specific middleware. A major challenge of current BPM solutions is to continuously monitor ongoing activities in a business environment and to respond to business events with minimal latency. One of the most promising concepts that approaches the problems of closed-loop decision making and the lack of gaining real-time business knowledge is the concept of Complex Event Processing (CEP), The system automatically discovers and analyzes business situations or exceptions and can create reactive and proactive responses, such as generating early warnings, preventing damage, loss or excessive cost, exploiting time-critical business opportunities, or adapting business systems with minimal latency. Claire Costello [4] says A business process as a complete set of end-to-end activities that together create value for the customer. Business process management is the ability to orchestrate and control the execution of a business process across heterogeneous systems and allow users to view the components of an infrastructure from a process view rather than set of applications and databases. Anca Andreescu [5] says every organization operates according to a set o business rules. These may be external rules, coming from legal regulations that must be observed by all organizations acting in a certain field, or internal rules which define the organization's business politics and aim to ensure competitive advantages in the market. Starting from the previous observations, it is obvious the important role that business rules play within the development process of a software system. Milan Milanović1 [6] proposed that BPM languages have limited support for representing logical expressions, business vocabularies, and business rules, which severely limits their flexibility and expressivity. To address these challenges, he integrated business rule modeling constructs of the REWERSE Rule Markup Language (R2ML) with the Business Process Modeling Notation (BPMN), resulting in a *rBPMN* proposal. Olegas Vasilecas [7] says Business rules make an important and integral part of each information system (IS) by expressing business logic, constraints of concepts, and their interpretation and relationships. Therefore it is relevant to pay special attention to business rules in development of information systems. Rules related to domain structure and behavior are presented in data, states, processes and other IS models. Taking into account that rules are expressed in several models, there is a risk that overall specification is inconsistent. Unambiguous models are crucial for the successful implementation of IS models transformation and finally code generation tasks. Therefore it is necessary to check consistency among related rules models. The problem of models inconsistency can be solved by using of formal or partially formal models with constraints. However formal models are often too complex to be used in practice. Semi-formal models are widely used, but constraints used in such a models often are suitable only for one model and relationships among models are not defined. He suggests extending of IS approach based on semi-formal models and constraints, by adding the consistency rules for IS models. Anis Charfi [8] applied the divide and conquer principle to web service composition by explicitly separating business rules from the process specification. The combination of the business rules approach with the process-oriented composition solves a twofold problem. First, he provided a solution to the problem of dynamic adaptation of the composition. In fact, current standards for process-based web service composition are not capable to deal with the flexibility requirements of composite web services. Second, business rules are important assets of a business organization that embody valuable domain knowledge. So, it is no longer acceptable to bury them in the rest of the composition. Bruno de Moura Araujo [9] proposed Business rules (BR) are declarations which constrain, derive and give conditions for existence,

representing the knowledge of the business. BRs are not descriptions of a process or processing. Rather, they define the conditions under which a process is carried out or the new conditions that will exist after a process has been completed. BR can be represented using Semantics of Business Vocabulary and Rules (SBVR). SBVR is appropriate to be used by business experts, since it allows the representation of business vocabulary and rules using controlled natural language. Business vocabulary concepts can be automatically transformed into conceptual models, like the UML class model. Nicholas Zsifkov [10] says Enterprise business rules are usually defined as constraints or as metadata about business operations: on the business side, business rules are special policies that define constraints/metadata about the business operation; on the information system (implementation) side, business rules are constraints about the data, about data manipulation and about system processes. Antonio Oliveira Filho [11] proposed a new traceability technique that defines dependency links with the same semantics that can be observed in the relationships among business rules. The goal of the technique is to provide rates of recall and precision of 100% for changes in software requirements that correspond to business rules. Jose F. Mejia Bernal [12] says decomposing the initial business process structure in a set of rules is a procedure based on pattern identification. This approach consists of two phases: mapping of business process to rules, and reliable adaptation of the business process according to the context data. The first phase is executed to provide a representation of the initial business process definition in terms of rules. The second phase is applied to provide a reliable workflow process modification.

Timon C. Du and Hsing-Ling Chen propose an active collaboration and negotiation framework (ACNF), which is a negotiation support system that uses active documents with embedded business logics or business rules that can adapt to different collaborative strategies in a business-to-business (B2B) environment [13]. Sam Weber and Isabelle Rouvellou describe a fully functional prototype middleware system which provides the users to control software components without the need of programming knowledge. Thus the core applications need not be altered for anticipated changes from external factors. In this system, application behavior modification is fast and easy, making this middleware suitable for frequently changing programs [14]. H. M. Sneed[15], in the context of reverse engineering source code into UML diagrams many tools and approaches have been developed.CPP2XMI is a reverse engineering tool which lows extracting UML class, sequence and activity diagrams in XMI format from C++ source code. Sangseung Kang[16], Business rules are business statements that define some aspect of a business. They describe, constrain and control the structure, operations and strategy of the business. In his paper, he analyzes business rules and business rule systems, and present requirements and considerations for the business rule expression and system. Olga Levina[17], suggests an extraction process for business rules identification from business process models. Applying this process introduces a structured approach and management aspects within rules discovery by focusing on rule sources that are important for the process goal and providing a rule structure. Mohammed Alawairdhi[18], suggests a business-logic-based framework for evolving software systems is proposed. The goal of the framework is evolving software in a higher abstract layer. Olegas Vasilecas[19], suggests The rules are expressed in several models, there is a risk that overall specification is inconsistent. Unambiguous models are crucial for the successful implementation of IS models transformation and finally code generation tasks. Therefore it is necessary to check consistency among related rules models. The problem of models inconsistency can be solved by using of formal or partially formal models with constraints. According to Anis Charfi[20], the Business Rules Group , a business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control the behavior of the business. Business rules are usually

expressed either as constraints or in the form if conditions then action. The conditions are also called rule premises. The business rule approach encompasses a collection of terms (definitions), facts (connection between terms) and rules (computation, constraints and conditional logic). Terms and Facts are statements that contain sensible business relevant observations, whereas rules are statements used to discover new information or guide decision making. Jose F. Mejia Bernal[21], proposed a way for representing Dynamic Business Process in terms of Rules based on patterns identification. With this approach it is easy to apply on a business process instance both user-based personalization rules and automatic rules inferred by an underlying context-aware system. Gulnoza Ziyaeva proposed framework to enable the content-based intelligent routing path construction and message routing in ESB which defines the routing tables and mechanisms of message routings and facilitate the service selection based on message content [22].

5. EXISTING SYSTEM

In existing system the Business Rules can be changed or modified only during the inert time period. So the main aim of this Project is to provide Business Rules that can be changed at the Run Time. Here the existing rules are modified and can be attached to source code without hindering service to the end user which can be achieved with source control systems. Once the services have been identified, the rules can be extracted and service can be provided dynamically. To provide this service we are going to make use of JESS language.

6. *BRMS architecture*

6.1. Rule capturing

Rule isolation and extraction is performed by tracing logic paths based on various selection criteria. This includes logic leading to the creation of a given output variable, logic linked to some type of conditional and logic associated with a given input transaction type. Analysts must review a rule after extraction in order to use it as is, extract again based on a different criteria or subset the extracted rule further. These techniques are somewhat tool dependent, but it is important to establish an initial extraction criteria regardless of the tool being applied.

Required rule extraction tool criteria minimally includes an ability to "slice" out a rule based on a specified selection criteria. Since business rules do not limit themselves to the confines of a source program, extraction tools must be able to analyze logic across program boundaries. Beyond that, a rule extraction tool should be able to bypass or highlight implementation dependent logic, store an extracted rule, further extract against a previously extracted rule, display a rule in varying formats to promote understandability and transform an extracted rule into a reusable format. Certain tools can load a system into its knowledge base and transform it into predicate logic. Rule analysis is then facilitated through cross system extraction and simplification techniques. Rules may be displayed in decision vectors or as source code. The real power of are that they allow IT professionals and non-technical analysts, to examine extracted rules and verify what functions a system is actually performing. While most tools offer static source code analysis to support rule extraction, other tools provide dynamic analysis by tracing logic paths during program execution.

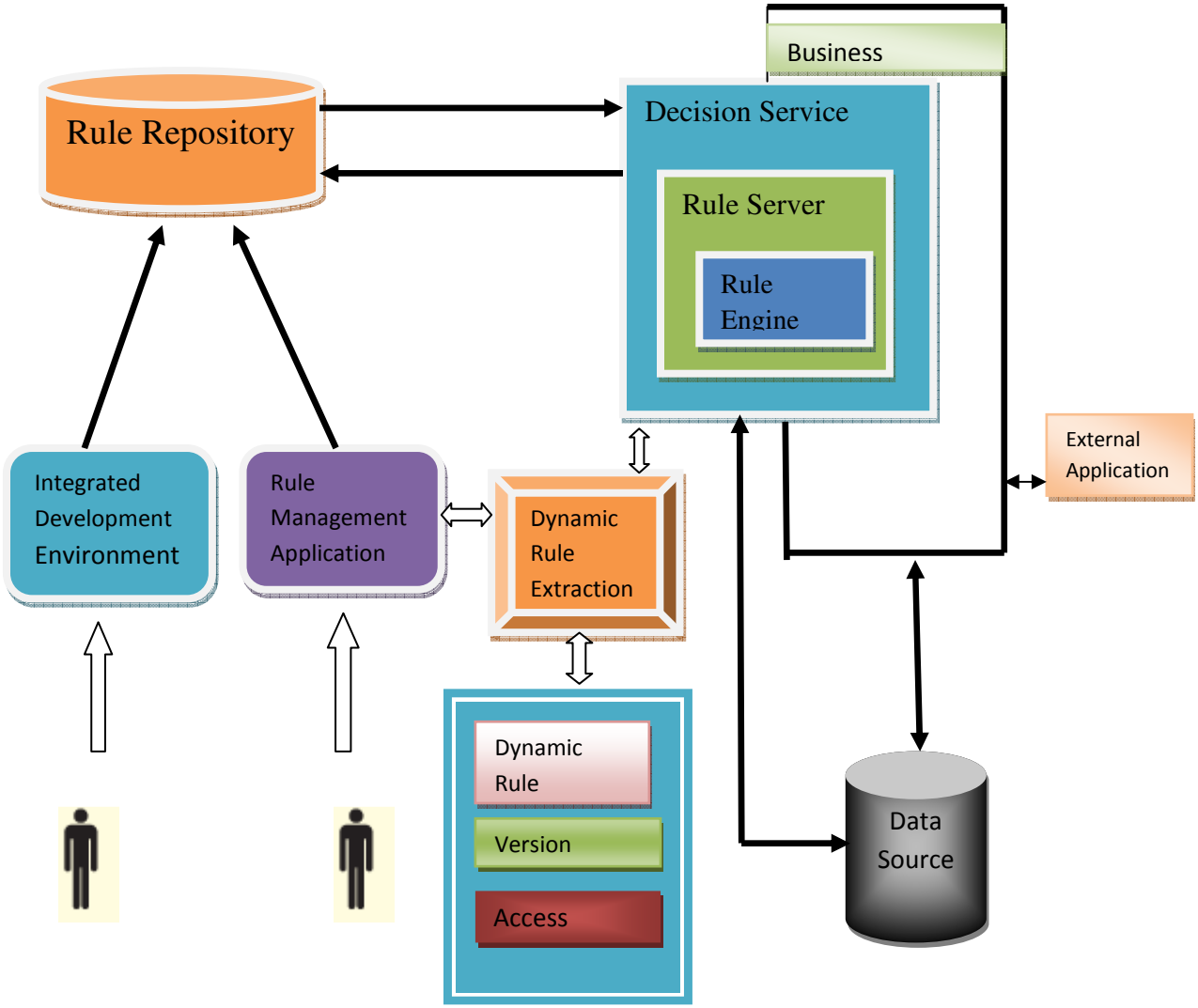

Figure 1

Figure 1. BRMS Architecture

Dynamic analysis is performed when a program is executed, dynamic analysis can capture a logic path based on the transaction being performed. Sophisticated tools support the creation of a union of slices allowing analysts to fine tune extracted views.

6.2. Rule Repository

A business rule repository is a system in which company uses to document, update, and keep track of the business needs and rules regarding your projects. Having a central repository to store these rules will allow developers and business owners access to rules, and any questions regarding business projects.

6.3. Decision service

A decision service is some kind of software component that acts as a business logic black box: other parts of a system present it with data, it makes potentially-complex business 'decisions' and returns some result. Typically, this is a component of a service-oriented architecture that encapsulates the business logic required to make business decisions, and which is called by applications that do not contain this logic themselves. In such an architecture, this communication typically uses web services. The first thing a Decision Service does is isolate the logic behind these business decisions, separating it from business processes and the mechanical operations of procedural application code. Treating decision logic as a manageable enterprise resource in this way means you can reuse it across multiple applications in many different operational environments. A centralized approach to decision automation can also eliminate the time, cost and technical risk of trying to reprogram multiple individual systems simultaneously to keep up with changing business requirements. For instance, the rules for paying an insurance claim can be removed from the definition of the claims processing business process. These rules can be managed independently, important as the legislative change cycle is different from the business cycle that drives process change. They can also be re-used, for instance to help customers tell if they have a valid claim before submitting it or to support third-party agents. The same rules can be in multiple decision services.

7. BENEFIT IN USING BRMS

Generally the rule capturing process focuses on the identification of the potential business rules sources. As business logic requirements change, business analysts can update the business logic without enlisting the aid of the IT staff. The new logic is immediately available to all client applications. All these process are done in run time of any real time application. This approach can be implemented with JESS language. Since JESS has an advantage of changing the business rules at the run time. First the rule which is to be changed is checked with the rules which are in rule repository. If the rules are already present in the repository means the rules are changed or modified for the particular application. Suppose the rules are not in the rule repository then it provides an option of adding new rules to rule repository. Based on this technique the following process is carried out. Here there are some examples of business rules which had been changed to JESS.

Algorithm for converting Business Rule to JESS

```
Procedure
Convert business rules→ Jess
def rule(rule)
get the rule
rule ← string defining the business policy
Convert the rule into java expression
Split the expression into operators, operands, parenthesis and store it into array
        arr[] ← split(operator, operand, parenthesis)
Backtrack the array and assign the elements to a string array
        for  i ←  n to 1 & j ← 1 to n  do
                element ← arr[i]
                str[j] ← element
        end for
```

```
pass the array str[] to the function convert
        input[] ← str[]
function convert(input[])
do
    //Read the next element.
        e ← input.next()
    if  e is operand        then
        outstr ← e
    end if
    if e = )    then
        stack.push() ← e // Push the operator on stack
    end if
    do
        if e is an operator then  // stack is empty, push operator on stack.
        if stack = empty   then
        stack.push() ← e
    end if
        if  stack.top ← )   then
        stack.push() ← e
        end if
        if e.priority == 0     then
        stack.push() ← e
        else
        temp ← stack.pop()
        outstr ← temp
        end if
    while (operator)
      if  e is  (    then  // a opening parenthesis, pop operators from stack
        if stack.pop() == )             then
        discard;
        else
        outstr ← stack.pop()  //  add them to output string
        end if
    end if
        If there is more input go to step 2
while(input.next == null)
do
        outstr ← stack.pop()
while(stack == empty)
JessExp ← reverse(outstr) //Reverse the string
end function
end procedure
```

8. BUSINESS RULE AND JESS SYNTAX

8.1. Rules For Car Licence

Table 1. Rules for Car Licence

S.NO	Rule	Syntax	JESS Syntax

1	Car must not be rented to customers without a valid license number.	if(Customer.ValidLicenceNumber == "FALSE")
		{
		application.Status = "Reject";
		Customer.Eligibile = false;
		}
			(def rule no car rented without lenience no)
			(if(=(Customer.ValidLicenceNumber ? true)
			=>(add(application.status)(/?reject)
			(add(customer.eligible)(/?false)

2	Car must not be rented to customers of Age less than 18.	if(Customer.age < 18)
		{
		application.Status = "Reject";
		Customer.Eligibile = false;
		}
			(def rule no car rented to customer of age less than 18)
			(if(=<(customer.age ? 18)(age?age)
			=>(add(application.status)(/?reject)
			(add(customer.eligible)(/?false)

3	Car must not be rented to customers with bad history level 3	if(Customer.BadHistoryLevel == 3)
		{
		application.Status = "Reject";
		Customer.Eligibile = false;
		}
			(def rule no car to customer of bad history level 3)
			(if(=(Custome.badhistorylevel ? 3)
			=>(add(application.status)(/?reject)
			(add(customer.eligible)(/?false)

4	Rent for small cars is 80 aud per day	if (Customer.Eligible == true)
		{
		if(Car.type == Small)
		{
		rent.RentPerDay = 80;
		}
			(def rule rent for small car is 80/day)
			(if(=(Customer.Eligible? true)
			{
			if(=(cartype?small)
			{
			=>add(rent.perday)(/?price 80))
			}

5	Rent for awd cars is 100 aud per day.	if(Car.type == AWD)
		{
		rent.RentPerDay = 100;
		}
			(def rule rent for awd cars 100/day)
			(if(=(car.type?AWD)
			{
			=>add(rent.perday)(/?price 100)
			}

6	Rent for luxury cars is 150 aud per day	if(Car.type == Luxury)
		{
		rent.RentPerDay = 150;
		}
			(def rule rent for luxury cars 150/day)
			
			(if(=(car.type?luxury)
			{
			=>add(rent.perday)(/?price 150)
			}

7	Rent payable is calculated as the product of rentperday and rentalperiod in days.	{
		rent.RentPayable = rent.RentPerDay * rent.No of rent days;
		if (CustomerBadHistoryLevel > 0)
		}
			(def rule rent calculated as the product of rentperday and rentalperiod in days)
			
			(rent.rentpayable)(=(*(?rentperday?rentalperiod)))
			(if(>(Custome.badhistorylevel ? 0)
			=>(add(rent.rentpayable)(price?price)

8	Penalty of 20 % of rent must be applied for customers with bad history level 2.	if(Customer.BadHistoryLevel == 2
		{
		rent.PenaltyFee = rent.RentPayable * 0.2;
		}
			(def rule Penalty of 20 % of rent for customers with bad history level 2)
			
			(if(=?Customer.BadHistoryLevel?2)
			{
			=>
			add(rent.penaltyfee)(=(*(?rent.rentpayable?0.2)
			))

9	Penalty of 10 % of rent must be applied for customers with bad history level 1.	if(Customer.BadHistoryLevel == 1)
		{
		rent.PenaltyFee = rent.RentPayable * 0.1;
		}
		}
			(def rule Penalty of 10 % of rent for customers with bad history level 1)
			
			(if(=?Customer.BadHistoryLevel?1)
			{
			=>
			add(rent.penaltyfee)(=(*(?rent.rentpayable?0.1)
			))

10	Total amount payable is calculated as the sum of rent payable and penalty if any.	{
		rent.TotalAmountPayable = rent.RentPerDay + rent.PenaltyFee;
		}
		else
		{
		
		}
			(def rule Total amount payable as the sum of rent payable and penalty.)
			
			{
			(rent.TotalAmountPayable)(=(+(?
			rent.RentPerDay? rent.PenaltyFee)))
			}
			=>(add(rent.TotalAmountPayable)(price?price)

8.2. Rules for Banking

Table 2. Rules for Banking

S.NO	Rule	Syntax	JESS syntax

1	1. A person is defined high if atleast one of the following is true:
	• The person should be VIP.
	• The person should have current balance of $500.
	• The person should have account for more than 5 years.
		If(bal> 500 && year > 5)
		{
		Print("the person is VIP")
		}
			(def rule person current balance >500 and account more than 5 years)
			
			(if(>(bal?500))
			{
			(if(>(year?5))
			=>(add(the person is VIP))(VIP?VIP)

2	A withdrawal from an account may be made only
	• If the is active and
	• Account balance should be greater than zero.
		If(acc=active)
		{
		If(bal>0)
		{
		Printf(" withdrawal is made")
			(def rule withdrawal is done if account is active and balance is greater than zero)
			
			(if(=(acc?active))
			{
			(if(>(bal?0)
			{
			=>add(withdrawal is made)(/ money?money)

3	Income Rule	if((basic sal && other income)>0))
		{
		income status =valid;
		}
			(def rule income)
			
			(if(>(?basic sal?other income?0)
			=>add(income status)(=?valid)

4	Commitment status rule	if (ID=ID)
		{
		if(creditcardbal>500)
		{
		print (" compute commitments")
		}
			(def rule commitment)
			
			(if(=(ID?ID))
			{
			(if(>(creditcardbal?500))
			{
			=>(add(compute commitments))
			}

5	Employment status rule	if(Employmenttype && months>18)
		{
		employment status=valid
		}
			(def rule employment status)
			
			(if(>(?employmenttype?months?18)))
			{
			=>add(employment status)(=?valid)
			}

6	Residency status rule	if(place of residence && time in months>18)
		{
		Residency status=valid;
		}
			(def rule residency status)
			(if(>(?place of residence?time in months?18)
			=>add(residency status)(/?valid)

8.3 Ticket Reservation

Table 3. Rules for Ticket Reservation

S.NO	Rule	Syntax	JESS syntax

1	The User must have to login .	If(user.acctno==”false”)
		{
		User.login=invalid;
		}
			(def rule user login)
			(if(=(user.acctno?false)))
			=>add(user.login)(=?invalid)

2	The available seats in the train should be visible to the user.
	• Provide Date, Time (arrival time, departure time).
		If(user.acctno=valid)
		{
		User.date=”journey date”;
		User.atime=”arrival time”;
		User.dtime=”depature time”;
		}
			(def rule for providing date, arrival time, depature time)
			(if(=(user.acctno?valid)))
			=>add(user.date) (=?journey date)
			add(user.atime) (=?arrival time)
			add(user.dtime) (=?depaturetime)

3	If the required train is available book the ticket.	If(user.train==”available”)
		{
		User.ticket=”true”;
		}
			(def rule for booking ticket)
			(if(=(user.train?available)))
			=>add(user.ticket)(=?true)

4	The ticket must contain details such as
	Id number
	• Name
	• No. of passengers
	• Seat no
	• Coach no
	• Class (I,II,III)
		If(user.ticket==”true”)
		{
		User.ticket.id=”id number”;
		User.ticket.name=”name”;
		User.ticket.passenger=”no.of passenger”;
		User.ticket.seat=”seat no”;
		User.ticket.coach=”coach no”;
		}
			(def rule for ticket details)
			(if(=(user.ticket?true))
			{
			=>add(User.ticket.id)(=?id number)
			=>add(User.ticket.name)(=?name)
			=>add(User.ticket.passenger)(=?no.of passenger)
			=>add(User.ticket.seat)(=?seat no)
			=>add(User.ticket.coach)(=?coach no)

5	The some amount of cancellation charge must be deduced from the user.	If(ticket.status==”cancelled”)
		{
		User.account=amount-20;
		}
		If(ticket.status==”cancelled”){
		If(user.account==”deduced”)
		{
		Return balance;
		}
		}
			(def rule for cancelling ticket)
			(if(=(ticket.status?cancelled)))
			{
			(User.account)(=(-(?amount?20)
			}
			(if(=(ticket.status?cancelled)))
			(if(=(user.account?deduced)))
			{
			=>add(return balance)(price?price)
			}

9. Experimental Approach

The Extraction of business rules can be created using .net framework. For storing the Business rules we are making use of MS access database. The Business rules for the several applications are first identified. If the rules are to be changed, here provide an option of editing the rule and store it in the repository. Suppose if the rules are already present in the repository it can be mapped and extracted otherwise new rules can be added to it. Moreover additional functions are added to this framework. The functions such as modify, update, extend. The application we developed is shown below.

9.1. Screenshots

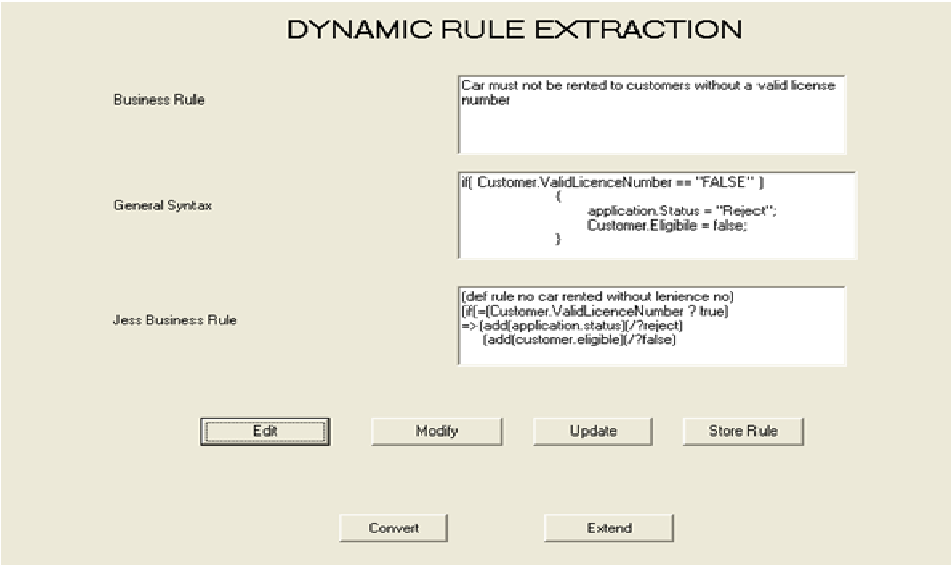

Figure 2. Dynamic rule Extraction

10. Conclusion

Changing the business rules dynamically, with the help of Jess language, has the purpose of providing flexibility, correctness and consistency. This paper provides a reliable and more flexible approach to handle dynamic changes in business rules. The Rule extraction can be made in two phases: first, the business rules are identified for the particular application is represented in terms of general syntax. Then the rules are converted into Jess syntax, in order to provide more flexibility when dynamic changes are made. Next, the business rules instance requires checking whether the modification will maintain the process consistency. There are many possible extensions of the presented work; we intend to extend the dynamic changes in business rules.

11. References

[1] Chun Ouyang , “From Business Process Models to Process-Oriented Software Systems”

[2] Michael zur Muehlen, “Business Process and Business Rule Modeling Languages for Compliance Management: A Representational Analysis”, Twenty-Sixth International Conference on Conceptual Modeling - ER 2007 - Tutorials, Posters, Panels and Industrial Contributions, Auckland, New Zealand.

[3] Josef Schiefer, "Event-Driven Rules for Sensing and Responding to Business Situations".

[4] Claire Costello, " Orchestrating Supply chain Interactions using Emerging Process Description Languages and Business Rules".

[5] Anca Andreescu, " A General Software Development Process Suitable for Explicit Manipulation of Business Rules"

[6] Milan Milanović1, " Modeling Service Orchestrations with a Rule-enhanced Business Process Language"

[7] Olegas Vasilecas," Ensuring Consistency of Information Systems Rules Models".

[8] Anis Charfi, " Hybrid Web Service Composition: Business Processes Meet Business Rules".

[9] Bruno de Moura Araujo, " A method for Validating the Compliance of Business Processes to Business Rules".

[10] Nicholas Zsifkov, " Business Rules Domains and Business Rules Modeling".

[11] Antonio Oliveira, " Filho Change Impact Analysis from Business Rules".

[12] Jose F. Mejia Bernal, " Dynamic Context-Aware Business Process: A Rule-Based Approach Supported by Pattern Identification".

[13] Timon C. Du and Hsing-Ling Chen ,"Building a Multiple-Criteria Negotiation Support System", IEEE transactions on knowledge and data engineering, vol. 19, no. 6, June 2007.

[14] Sam Weber, Hoi Chan, Lou Degenaro, Judah Diament, Achille Fokoue-Nkoutche, and Isabelle Rouvellou, "Fusion: A System For Business Users To Manage Program Variability", IEEE Transaction on software engineering , Nov 2008.

[15] H. M. Sneed, "Extracting Business Logic from Existing COBOL Programs as a Basis for Redevelopment", 9th International Workshop on Program Comprehension, Toronto, Canada, 2001, pp. 167-175.

[16] Sangseung Kang, "Design of Rule Object Model for Business Rule Systems", 1996,pp. 818-822.

[17] Olga Levina, "Extracting Business Logic from Business Process Models", 2010 IEEE.

[18] Mohammed Alawairdhi, "A Business-Logic Based Framework for Evolving Software Systems", 2009 33rd Annual IEEE International Computer Software and Applications Conference.

[19] Olegas Vasilecas, "Ensuring Consistency of Information Systems Rules Models", the International Multiconference on Computer Science and Information Technology, 2008.

[20] Anis Charfi, " Hybrid Web Service Composition: Business Processes Meet Business Rules", ICSOC'04, November 15–19, 2004, New York, New York, USA.

[21] Jose F. Mejia Bernal, " Dynamic Context-Aware Business Process: A Rule-Based Approach Supported by Pattern Identification", SAC'10, March 22-26, 2010, Sierre, Switzerland.

[22] Gulnoza Ziyaeva, Eunmi Choi, and Dugki Min, "Content-Based Intelligent Routing and Message Processing in Enterprise Service Bus", International Conference on Convergence and Hybrid Information Technology, 2008

[23] *en*.wikipedia.org/wiki/Business_rule

Authors

Thirumaran. M, working as Asst.Professor in Pondicherry Engineering College, Pondicherry, India, one of India's premier institutions providing high quality education and a great platform for research. He pursued his B.Tech and M.Tech in Computer Science and Engineering from the Pondicherry University. The author is specialized in Web Services and Business Object Model and possesses a very profound knowledge in the same. He has worked on establishing the Business Object Model and Business Logic System with respect to Web Service Computation, Web Service Composition and Web Service Customization. His flair for research has made him explore deep in this domain and he has published more than 20 papers in various International Conferences, Journals and Magazines. Currently he is working on developing a model for Business Logic Systems for various E-Commerce systems.

Ilavarasan. E, working as Associate Professor in Pondicherry Engineering College, Pondicherry, India. He received his Bachelor's degree in Mathematics from the University of Madras in the year 1987 and Master's degree in Computer Applications in the year 1990 from Pondicherry University. Later he completed his M.Tech., degree in Computer Science and Engineeering at Pondicherry University in the year 1997. He has published more than Twenty five research papers in the International Journals and Conferences. His area of specialization includes Parallel and Distributed Systems, Design of Operating System and Web Technology.

Thanigaivel. K, studying in Pondicherry Engineering College, Pondicherry, India. He received his B.Tech degree in Computer Science and Engineering from the Pondicherry University in the year 2009. He currently pursuing M.Tech., degree in Computer Science and Engineering at Pondicherry University and he is currently working in the area of Webservices.

Abarna. S, studying in Pondicherry Engineering College, Pondicherry, India. She currently pursuing M.Tech., degree in Computer Science and Engineering at Pondicherry University and she is currently working in the area of Webservices.